UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Implementation of an evaluation platform for Alzheimer patients based on Egocentric Sequences Description

Author: Sergi Soler Solé

Director:	**Marc Bolaños**
Co-Director:	**Petia Radeva**
Done in:	**Departament de Matemàtiques i Informàtica. UB**
Barcelona,	**30 de gener de 2017**

Abstract

Numerous international population-based studies have been conducted to document the frequency of MCI, estimating its prevalence to be between 15% and 20% in persons 60 years and older, making it a common condition encountered by clinicians[17]. This number is predicted to increase to 75.6 million in 2030, and 135.5 million in 2050[14], leading to deep social and economical costs. The most common dementia type is Alzheimer (between 50% and 70% of the cases) and its early detection can greatly affect the recovery of the patient. That is why it is important to have tools for its early diagnosis and follow-up.

Serious games, with an increasing popularity, are a good way to MCI as an early stage of Alzheimer and improve the memory capacities of the patients. These video games focusing on different stages of the illness can help doctors to document and check the progress of the illness.

This work aims on developing a software for patients with MCI, which is the lack of memory and other human characteristics like reasoning and language. These individuals usually progress to Alzheimer disease, but if detected early, in some cases they can also remain stable or even recover with time. To help them to exercise their memory, we propose that our program uses their own experiences caught by a wearable camera. This software will provide images of the patient's life in order to do exercises that will evaluate their ability to remember and reason about the scenes they visualize. With these tests, the doctors will be able to see the evolution of the patients, and help them to diagnose and track the illness.

In this project, we additionally work on an application, for the first time, of Deep Neural Networks for the automatic generation of descriptions of egocentric sequences. This will serve as the first step to automate the evaluation process by automatically comparing the subjective descriptions provided by the patients to the objective ones generated by our system.

Resumen

Diversos estudios internacionales de población , han documentado la frecuencia del Mild cognitive Impairment (MCI), estimando que su predominio está entre el 15% y el 20% en personas mayores de 60 años, haciendo que sea una enfermedad muy comuna encontrada por el personal médico [17]. Este número fue predicho que se incrementaría hasta los 75.6 millones en 2030, y 135.5 millones en 2050, lo cual provocaría unos costes sociales y económicos inmensos. La demencia mas comuna es Alzheimer (entre 50% y 70% de los casos) y su pronta detección puede afectar en gran medida en la recuperación del paciente. Por esto mismo es importante tener herramientas para su detección temprana.
Los juegos serios van ganado popularidad, estos video-juegos son una buena manera de diagnosticar el deterioro cognitivo leve y mejorar la memoria de los pacientes, estos juegos se centran en diferentes etapas de la enfermedad y pueden ayudar al personal médico a documentar y hacer un seguimiento de la enfermedad.
Este proyecto busca desarrollar un software para pacientes con deterioro cognitivo leve, que es la falta de memoria y otras características humanas como el razonamiento y el lenguaje. Estas personas pueden progresar al Alzheimer, pero tarda a tiempo puede permanecer estables o recuperase con el tiempo.
Para ayudar a ejercitar la memoria el programa usa sus propias experiencias recogidas con una camera portátil. Este software proporciona imagines de la vida del paciente con el objetivo de evaluar su habilidad para completarlos, con estos test los doctores serán capaces de ver la evolución de la enfermedad del paciente y ayudara a diagnosticar y seguir la enfermedad.

En este trabajo, también trabajamos en una aplicación, por primera vez, de Deep Neural Networks para la generación automática de descripciones para secuencias egocéntricas. Esto servirá como primer paso en la evaluación automática del proceso comparando automáticamente, la descripciones subjetivas del paciente con las objetivas creadas por el sistema.

Resum

Diversos estudis de població internacionals, han documentat la frequencia de Mild Cognitive Impariment (MCI), estimant que el seu predomini està entre el 15% i el 20% en persones de 60 anys i més grans, fent que sigui una condició molt comuna trobada per el personal mèdic[17]. Aquest número es va predir que incrementaria als 75.6 milions al 2030 i 135.5 milions al 2050, portant grans costos tant socials com econòmics. El tipus de demència més comuna es l'Alzheimer, entre (50% i 70% dels casos) i la seva ràpida pot afectar dramàticament la recuperació del pacient. És per aixo que és important tenir eines per la seva ràpida detecció.
Els jocs seriosos, amb una popularitat incremental, són una bona manera de diagnosticar persones amb MCI i millorar la capacitat mental i de memòria dels pacients, aquests videojocs es centren en diferents estats de la malaltia i poden ajudar al doctor documentar i revisar el progrés de la malaltia.
Aquest projecte busca desenvolupar un software per pacient amb Deteriorament Cognitiu Lleu, que es la falta de memòria i altres característiques humanes com el raonament i el llenguatge. Aquests individus normalment acaben progressant a la malaltia de l'Alzheimer, però si es detecta aviat, en alguns casos també pot romandre estable o fins i tot recuperar-se en el temps.
Per ajudar a exercitar la memòria, el programa utilitza les experiències del pacient, capturades amb una càmera portàtil. Aquest software proveirà imatges de la vida del pacient i les utilitzarà per crear exercicis per al pacient per avaluar la seva habilitat per completar-los, amb aquesta informació, el doctor serà capaç de veure la evolució dels pacients i ajudarà a diagnosticar i seguir la evolució de la malaltia.

En aquest projecte, també treballem amb una aplicació, per primera vegada, de Deep Neural Networks per la generació automàtica de descripcions egocèntriques. Això servirà com a primer pas per automatitzar el procés de avaluació comparant automàticament les descripcions subjectives generades per els pacients i les objectives generades per el nostre sistema.

Acknowledgments

I would like to thank Marc Bolaños for all the inestimable help and advice given at all times. I would also like to express my gratitude towards Dra. Petia Radeva for trusting me with this project and to all the medical staff that is working on finding a solution to improve the quality of life of the patients.

Contents

1 Introduction **1**
- 1.1 Problem . . . 1
- 1.2 Objectives . . . 2

2 State of the Art **3**
- 2.1 Serious games and application for dementia . . . 3
- 2.2 Egocentric vision . . . 3
- 2.3 Our proposal . . . 4

3 EgoMemory **5**
- 3.1 Technologies . . . 6
  - 3.1.1 Exercises and Evaluation . . . 6
  - 3.1.2 Memory enhancement . . . 7
- 3.2 Design . . . 8
  - 3.2.1 Exercises . . . 10
  - 3.2.2 Evaluation . . . 12
  - 3.2.3 Memory enhancement . . . 14
- 3.3 Architecture . . . 15
- 3.4 Implementation . . . 16
  - 3.4.1 Exercises and Evaluation . . . 16
  - 3.4.2 Memory enhancement . . . 18

4 Results and evaluation **20**
- 4.1 Exercises . . . 20
- 4.2 Evaluation . . . 20
- 4.3 Memory enhancement . . . 20
  - 4.3.1 Dataset . . . 21
  - 4.3.2 Metrics . . . 23
  - 4.3.3 Experimental results . . . 23
  - 4.3.4 Prediction results . . . 24

5 Conclusions **27**

Chapter 1

Introduction

1.1 Problem

Mild cognitive impairment (MCI) is a syndrome in which someone has problems with cognition and mental abilities such as memory or thinking. It represents the one of the earliest clinical stages of Alzheimer disease and other related dementia. MCI affects between 3% and 19% [6] of adults older than 65 years, and more than half of them progress to dementia within five years, although in some cases can remain stable or even return to normal over time. Thus, adults with mild cognitive impairment have high risk of progression to Alzheimer, which is a neurodegenerative illness whose symptoms are the loss of immediate memory and other mental capabilities. During this disease, nervous cells die and different parts of the brain start to fail. Gradually, bodily functions are lost, leading to the death of the patient. Therefore, diagnosing and tracking MCI at early stages of the illness is a big step at helping people suffering from it.

Serious games, are video games created not only for the entertainment of the user, but also for the learning or study of said users. Currently, serious games in medical environments are proliferating, specially the ones dedicated to people with mild cognitive impairment or Alzheimer. Over the last few years, several video games focused on different stages of progress of the illness, have been developed. The idea behind all these games is to entertain users while also trying to reduce and documenting the decline of the mental health capabilities of the patient and consequently, improving their living standards.

Egocentric images were used to create a serious game. These images consist of photos that capture the daily experiences of the user by using a wearable camera. Wearable cameras are small smart cameras that can be worn as an accessory; these cameras are basically used for lifelogging [18], but also can be used for leisure, sport, adventure. The advantage of egocentric images compared to other serious games, like memory exercises with pan and paper, is that the user finds these images more natural for them, (since looking at our photos is one of the most normal activity). So, the hypothesis of our project is that a cognitive framework based on re-living ow experiences through observing

episodic images will create a more efficient environment for the patient involved in the intervention.

The wearable camera used in our project is a Narrative Clip, which is Low Temporal Resolution (LTR) (2 or 3 frames per minute), because such cameras more suitable to acquire data over long periods of time. We set it to take a photo every 30 seconds, hence a day used to generate about 1500 images.

In this project, we seek to use egocentric images taken by mild cognitive impairment patients to automatize and systematize the evaluation and treatment of adults with MCI. To automate this process, a video description algorithm will be used to generate sentences from the egocentric images. These images are real experiences and memories of the patient suffering the illness, so when the patients watch the egocentric images, they will bring back memories which will help them to remember and relive those moments. Thus, with this project, we want to give a tool to the doctors to facilitate the acquisition of the state of the patient and enable to treat and decrease the progress of the illness.

1.2 Objectives

The main objective of this project is to create an image-based tool to help people who have mild cognitive impairment by making use of egocentric images. To join the different parts of the project (i.e. MCI, egocentric images and serious games), we seek to create an application that uses images of experiences lived by the patient to generate descriptions. These descriptions are generated using an algorithm of video captioning so that they can exercise their memory. The program will be used for doctors (also called evaluators, as they evaluate and treat the patients) as a tool to ease and automatize their job. In this work, there are two main parts to consider:

- Video description using egocentric images: there are already some works in the bibliography tackling the problem of video captioning, but they all use conventional videos or single images, and no work has been applied in a clinical environment. In our case, the algorithm will generate sentences from a sequence of egocentric images, which is much harder problem, because the egocentric frames are more spaced in time and the quality of the images is inferior due to the free movement of the wearable camera and the actions taken from a first person perspective.

- Treatment and evaluation platform: we need to create an interface for the serious game to facilitate the job of the evaluators. The interface needs to be easy to use and we have to take into account that not only the doctors, but also the patients will interact actively or passively with it, meaning that the buttons, text, images, etc. need to be appropriate to be easy to see and interact for the patient. There also needs to be a clear distinction on the parts that will be accessed by the patient and the ones accessed by the evaluator.

We also seek to join both parts automatizing the evaluation and treatment process as much as possible.

Chapter 2

State of the Art

2.1 Serious games and application for dementia

Due to the increasing lifespan of people, the occurrence of dementia has risen dramatically, making it one of the most common illnesses on elderly people. This trend has lead to a growing interest on serious games. Most of these games like Serious Game for Cognitive Testing of Elderly [7], try to detect dementia as early as possible implementing four test items computerized as one program. On the other hand, Kitchen and cooking [8] uses a cooking game to create emotions to the MCI patients, and uses video games as tools to evaluate and help to asses the patients' treatment, stimulation and rehabilitation. These games suffer from the inability of the elderly people to make a correct use of the games at their disposal, as the major part of them are not used to work with this kind of technologies; hence, these tests and games are not as natural and common for the patient. That is why we introduce egocentric images in the serious games, to make the experience as natural as possible for the patient.

2.2 Egocentric vision

The problem of video captioning recently appeared thanks to the new Deep Learning techniques. Most of these works make use of the classical encoder-decoder methodology used in the Machine Translation (MT) field. Newer approaches have been proposed, where the translation process is carried out by means of a large Recurrent Neural Network(RNN)[16]. The reintroduction of Deep Learning in the Computer Vision field through Convolutional Neural Networks (CNN) has allowed to obtain new and richer image representations.

Most of the works that generate textual descriptions from single images [9] also follow the encoder-decoder architecture, using a combination of CNN and Long Short-Term Memory (LSTM) on the encoding phase and an LSTM on the decoding part [1, 10, 11].

2.3 Our proposal

ABiViRNet [1] (Attention Bidirectional Video Recurrent Net for video captioning) is a recurrent Neural Network algorithm that tries to push further the combination of Convolutional and Recurrent Neural Networks by producing richer image representation and introducing Bidirectional Recurrent Neural Networks. We will train the dataset trained in [1] so it can generate the right sentences or description for egocentric images. This project will be the first to present a dataset for video description composed of egocentric images.

Regarding the treatment and evaluation platform of the project, the initial part, consisting in the evaluation of the remaining cognitive capabilities of the patients, was already presented in [2]. In this work, we present the extension and improvement of the previously presented part, where we focus on the initial part of the treatment procedure. The doctors will have ten episodes (repeating the same tests in ten different occasions), to check the progress of the mild cognitive impairment disease of the patient.

Chapter 3

EgoMemory

Our proposal, which we call EgoMemory can be divided in three blocks. In Figure 3.1, we can see the design of our proposal divided in the three blocks and the communication between them:

Exercises: this is the part of the platform dedicated to the patients suffering from mild cognitive impairment and where they will have to do a serious game with the presence of an evaluator that will ask questions and check their reactions.

Evaluation: it is the part of the platform that will serve as evaluation for the patient. In this part, the doctor can see and analyze the results of the serious game done by the patient and from those results create subjective descriptions of a video created from egocentric images and check on the patient's feelings and emotions.

Memory enhancement: this last part is not related to the interface, it is the one responsible of the video description that will generate objective captions from a video created from egocentric images. To generate the objective captions, the video description algorithm needs to be trained first with a dataset composed of egocentric images and their associated description. In a future work, the description from the evaluation and from the memory enhancement parts will be automatically compared.

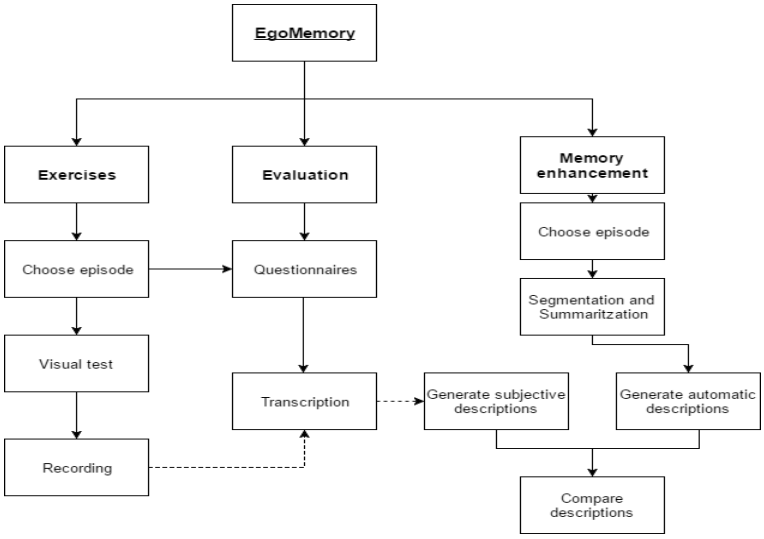

Figure 3.1: Structure of the proposal and its blocks

3.1 Technologies

3.1.1 Exercises and Evaluation

Technologies used on exercises and evaluation blocks are the same, as they both are a Graphical User Interface (GUI). These technologies have been chosen according to the needs of the doctors and what we analyzed that was the best to satisfy them.

At the hospitals, our clinical team works with the Windows operative system, so we have discarded other platforms like iOS and Android. For security and privacy matters, the program does not have to work on a network and neither needs internet connection. That is why we have decided to make it a desktop application and not a web one.

To program the system, we have chosen Java, since Java is a multiplatform language and could be used in other operative systems if the need arises. As a development editor, we have chosen Netbeans IDE given its easiness of use and complete graphic user interface (GUI) editor. As a database, we have used SQLite, taking into account that the database will be small and needs to be portable. Also, SQLite does not need internet connection, which was one of the requirements of the project and in general, SQLite is much faster than other databases like SQL.

To connect the database with the Java program, we have used a library called Hibernate. Hibernate is an ORM (Object Relational Mapping) that enables you to develop persistent classes following natural Object-oriented idioms, like inheritance, polymorphism, etc. Hibernate is also dependent of the language of the database, meaning that if we find us in the need of changing SQLite for any other database, we would not need to change any instruction in the code. Besides, Hibernate offers superior performance over straight JDBC code, both in terms of developer productivity and runtime performance.

During the development process, we found the need of using other external Java libraries. Those libraries are:

1. hibernate4-sqlite-dialect-0.1.2: This library is used to add SQLite dialect to Hibernate.

2. JavaFX: Used to implement audiovisuals.

3. SwingX: This library helps to create an interface with Java.

As the interface is the continuation of [2], we needed a program to control versions of the project, we chose GitHub with source tree as a git client. With source tree, you can visualize all branches of the project in a a graph, and mark all lines changed in the code since the last push. It also allows to easily stage or discard files in every commit.

3.1.2 Memory enhancement

Regarding the technologies used to train the video description algorithm, we used Python 2.7 as the development language. The main characteristic of Python is that it is easy to read. This easiness on reading helps you to think more clearly, when writing programs. It is also fast to write code, which helps to improve the productivity as the speed of writing increases.

To execute the algorithm, we also needed to install the Python library Theano, which helps us to define, optimize and evaluate mathematical expression involving multi-dimensional arrays. It also allows to use the GPU to perform data-intensive calculations.

Another library, we used is a modified version of Keras[1]. Keras[2] is a high-level neural networks library written in Python capable of running on top of Theano. This module of Keras, has the functionality to add learning rates multiplier to each of the learnable layers and has new layers for learning multi-modal and sequence-to-sequence problems.
In addition to the modified version of Keras, we used the Multi-modal Keras Wrapper[3] that gives support to easy multi-modal data and models loading and handling. Multi-modal Keras Wrapper consists on two basic components: The Dataset class, which stores, pre-processes and loads any kind of data for training a model (inputs) and the ground truth associated to the data (outputs). Also, it is in charge of loading the data in batches for training or prediction and the class Model_Wrapper that is in charge of storing an instance of a Keras Model, receiving the inputs or outputs of the Dataset class and using the model for training or production. Moreover, it provides two different methods for prediction(predictBeamSearchNet()[4] and predictNet()[5]). Lastly, we used COCO-Caption evaluation package that contains metrics to evaluate and compare the results of the model.

[1]https://github.com/MarcBS/keras
[2]https://keras.io/
[3]https://github.com/MarcBS/multimodal_keras_wrapper
[4]http://marcbs.github.io/multimodal_keras_wrapper/modules.html#keras_wrapper.cnn_model.Model_Wrapper.predictBeamSearchNet
[5]http://marcbs.github.io/multimodal_keras_wrapper/modules.html#keras_wrapper.cnn_model.Model_Wrapper.predictNet

3.2 Design

The developed interface is exclusive for the doctors and their patients, so the design is based on their requirements, while maintaining consistence with the parts of the program previously done. The program also needs to be clear and easy to use for a faster learning experiences and browsing through the windows of the application has to be intuitive. First of all, a Database was added, previously the information was saved in json files. We decided to change it to a database, because databases can handle very complicated queries that in a files system can be slow and time-consuming to do. Other benefits of the database in front of a files system are the multi-threaded access and the option to add an ORM to the database to manipulate the data in a very programmer-friendly way.

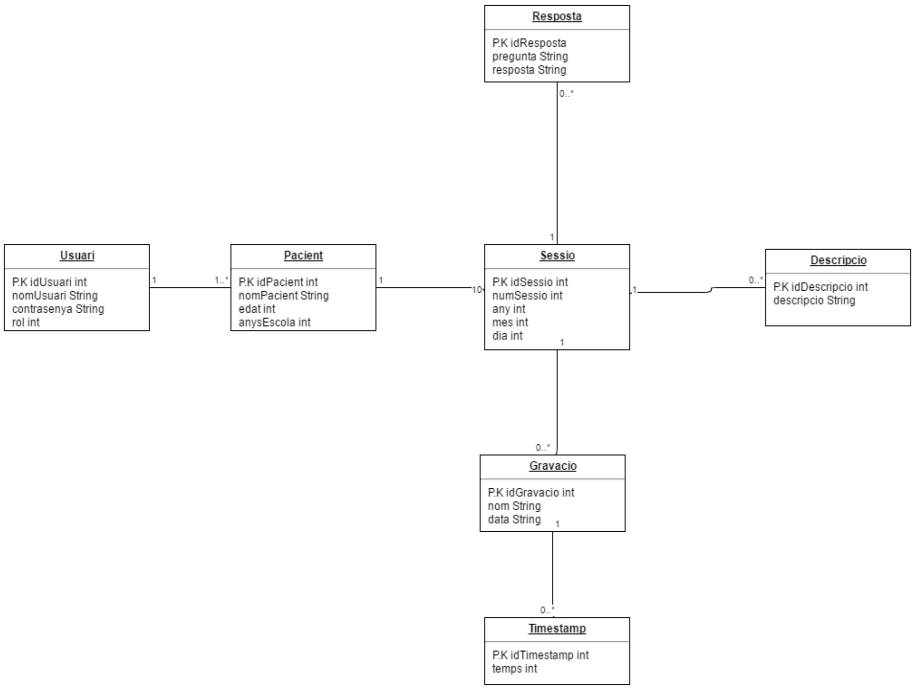

Figure 3.2: Database diagram

As we can see in the database diagram 3.2, our application will consist on users (doctors) that will be able to log inside and create patients. Each user will have a variable amount of patients. Then, each patient has ten episodes, an episode is a meeting with a doctor, where he or she will do the tests in the program, and in each one of these episodes all the data obtained during these tests are saved.

User Management

To gain some security, we had to create an authentication system that validates if the user is allowed to use the program.
There are two types of users:

- Administrators, once logged in the program, have the ability to create and modify the name and password of all evaluators (always checking that the name is not repeated) in the database. They have the ability to remove users from the system as well. Their only role is to administrate other users.
- Evaluators, who can manage patients, can create, remove or modify them. Moreover, they have access to all tests in the program.

Each user enters in a different menu upon logging.

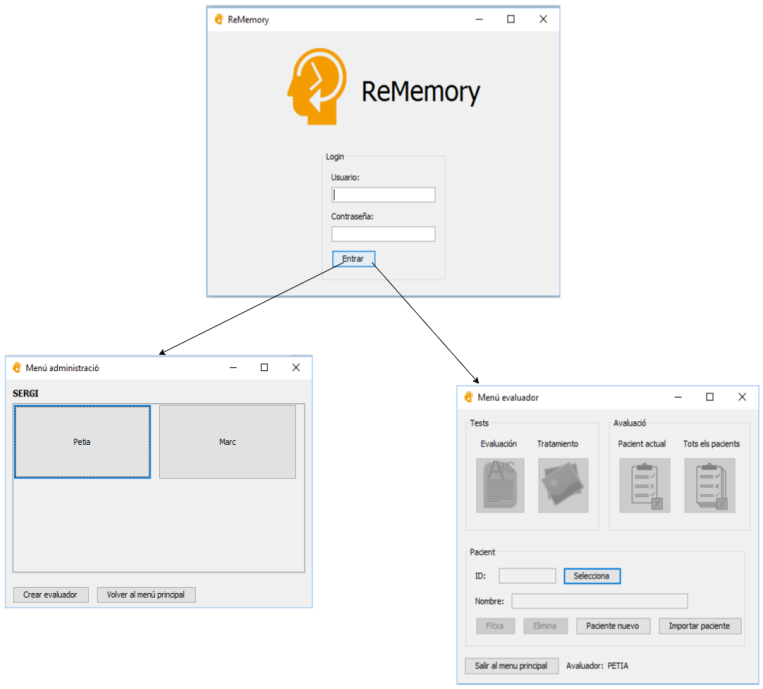

Figure 3.3: Different menus for different users, left is for administrators, right for evaluators

Patient Management

Regarding the patients, we save their name, id and the answers of all tests with the corresponding question. Once created, the patients can be selected from the evaluator menu to start working with them. Each patient has ten episodes, which all consist of analogous tests and questionnaires, and as being done, EgoMemory saves their answers in the database.

An episode is a sequence of images belonging to a certain timespan, in which the doctors considered that the patients were performing an activity that accomplishes some criteria: 1) being not routine, 2) there are people (preferably acquaintances), 3) the environments are dynamic, 4) enable certain feelings to the user, etc. These criteria are considered for the doctors as potential elements for enabling a faster progress of the MCI treatment.

Evaluators are responsible of creating and deleting patients from the database as well as modifying their profiles.

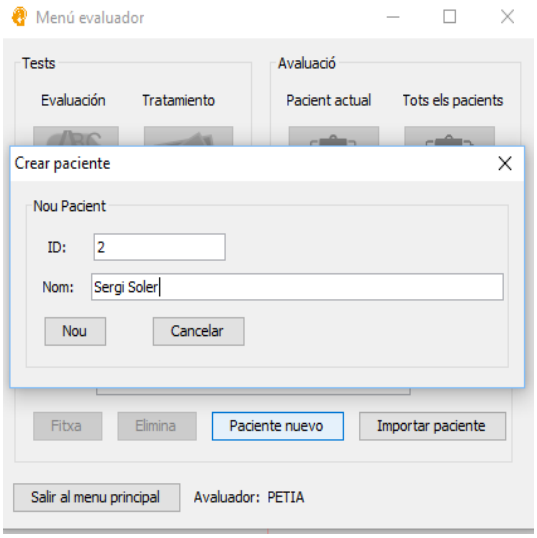

Figure 3.4: Window to create a new patient

3.2.1 Exercises

Visual test

From the evaluators menu, we can choose between two test options, *Evaluación* or *Tratamiento*[6] 3.3. The first one will open a menu with tests implemented in [2], which consists of tests to check the evolution of the MCI progress along time, while the second one will bring a menu that shows ten episodes available for the patient 3.5, and allows to enter in the visual test and other tests that will be implemented in a future. Each episode is a session with the patients' doctor. Each episode consists of a questionnaire at the start of the episode and one at the end and in between the different tests.

The visual test, which needs to be done by the patient with the presence of a doctor, will show a video created from the egocentric images 3.6. This video of past experiences of the patient will bring memories and emotions to the patient, which could help in their recovery. This window also has the option to create recordings. These recordings can be started and stopped at any time, each time creating a different file, and will serve to save what the patient is saying while not losing time making annotations. In the same window,

[6]Note that we use the label in Spanish following the requirements of our clinical team

there is a button to create time stamps to remember important times of the recording. This time stamp will be later used in the evaluation block to save time from the doctor finding the important parts of the recording.
This test will consist on the doctor asking questions to the patient, while the last one watches the video of his/her egocentric images and answers the doctor questions.

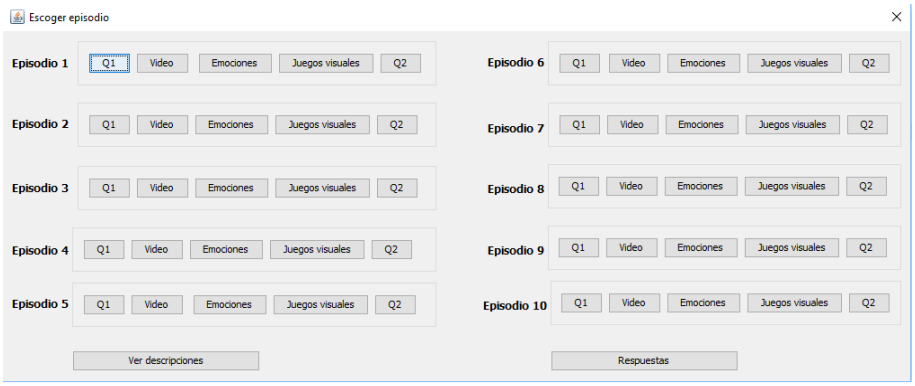

Figure 3.5: Window to choose a test from an episode

Figure 3.6: Visual test

3.2.2 Evaluation

This part of the program is intended to make an evaluation of the progress or feelings of the patient. It includes the subjective evaluation of the patient and the questionnaires that are made before and after each episode (Q1 and Q2 in 3.5).

Questionnaire

To evaluate the feelings and emotions of the patient, the doctor will make a test to the patient, who will have to answer different questions about how he/she felt before and after each episode. These answers are just a number between one and ten, being one not at all and ten a lot. This questionnaire will always have the same questions and will help the evaluator to guide the next episodes, as they will know if the test has affected them depending on their answers.

Figure 3.7: Questionnaire of emotions

Transcription

This window 3.8 will be used by the doctor after the intervention with the patient. It has the option to reproduce all recordings recorded in the visual test3.6. The doctor will be on full control of the recording, being able to start, stop, pause and move to the second wanted. The window also shows all time stamps created for the recording playing at that moment. Clicking on them will move the time bar to the seconds showed by the time stamp facilitating the job of the doctor of finding the crucial moments of the chosen episode. To help even more on the evaluation, at the bottom part of the window there is a space, where description of what happened can be written and viewed on the right.

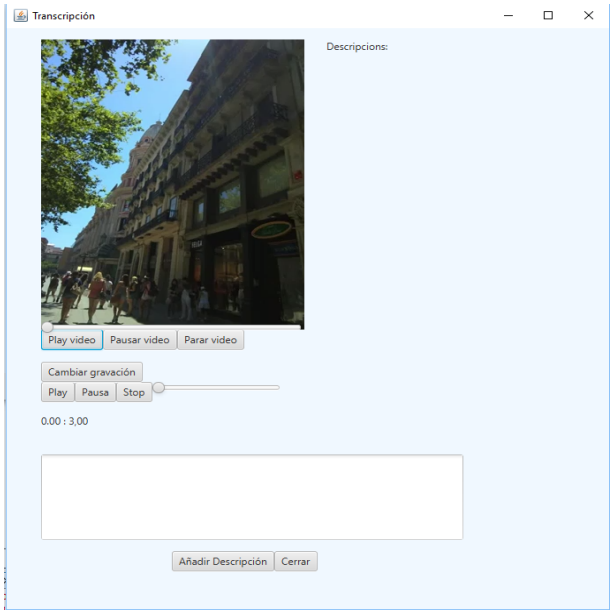

Figure 3.8: Transcription window

Results of evaluation

The results of the program, answers of the questionnaires3.7 and the descriptions introduced in 3.8 can be accessed from the main menu 3.5. At the bottom, there are two buttons, *Ver descripciones*, that opens a window showing all descriptions sorted by time and episodes, and the other button, *Respuestas*, that shows all questions and their answers answered by the patient. These are temporal windows. In the future, these results together with all tests implemented in [2], will be deployed in an excel file.

3.2.3 Memory enhancement

In order to achieve memory enhancement we claim to apply automatic video description algorithms in order to compare them in thefuture with the patient's narrative. More specifically, we gathered an egocentric dataset together with its associated textual descriptions. We used the model in [1] pre-trained on the Microsoft Research Video Description dataset (MSVD) [3] and applied a fine-tuning on the egocentric images. By using this model on the patients' images, we are able to obtain automatic descriptions that can be compared to the ones provided by the patients during the visualization of their episodes.

Episode selection, summarization and description generation

This is the part, where the video description algorithm will be applied, the doctor will choose an episode, consisting of a group of egocentric images forming a cohesive action, and a description will be created using our video description model. The limitation and difficulty of the egocentric images is that the data have a low temporal resolution (2 frames per minute) and there is little information in them, as the images are in first person and are also spaced in time. Then, a video will be created from those images using segmentation and summarization algorithms that will be shown to the patient in the visual test (Fig. 3.6). With this information, a subjective description will be created by the doctor and the patient, and another description of the video will be created from the algorithm. In future work, these descriptions will be compared and shown to the doctor.

3.3 Architecture

To organize the code, we used the pattern Model-View-Controller (MVC), as the project was started using this pattern, it helps to reuse the code and makes it more understandable. MVC is based on having these three parts with clear functions, the model stores data that are retrieved by the controller, the view generates the output of the software or creates the user interface, and the controller whose function is to send commands to the view and the model based on inputs from the users.

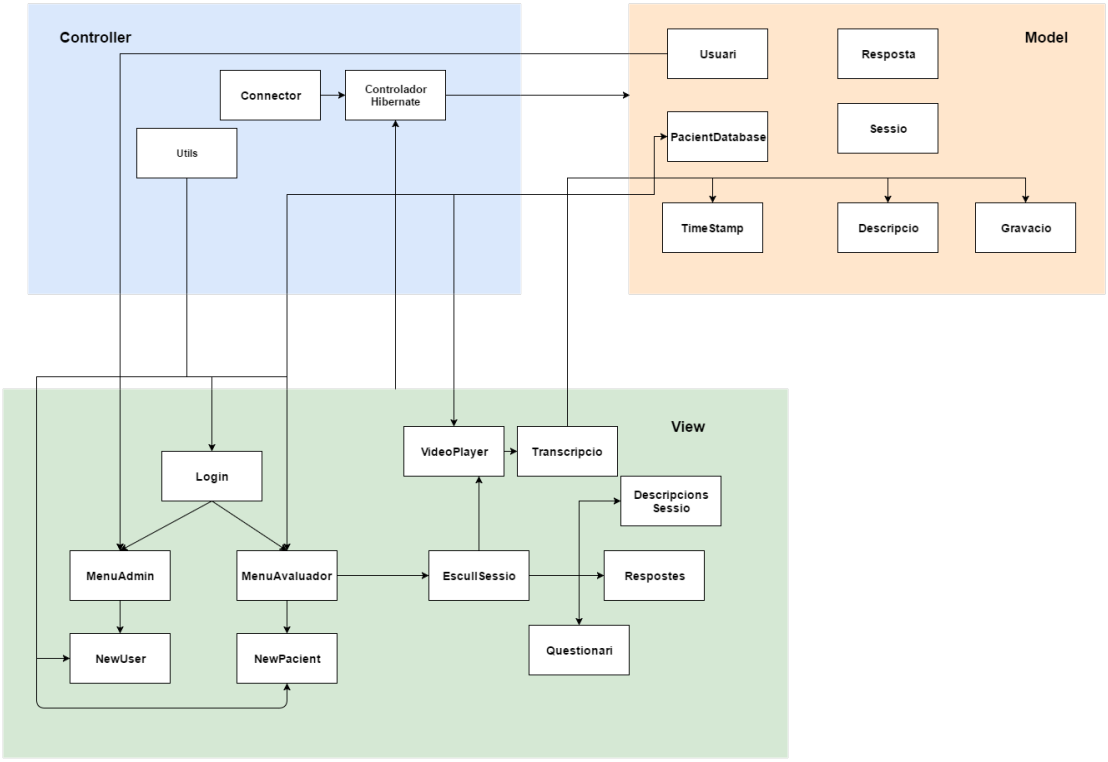

Figure 3.9: Class diagram with MVC pattern

Figure 3.9 shows the MVC pattern of the project, where there are only the classes created or modified in this project, although there are more classes already implemented by [2]. In controller (blue), the class connector is used for loading the configuration of Hibernate and to create sessions, which allows us to connect to the database. ControladorHibernate manages the access to the database; it has all the functions needed to read, create, modify and delete the tables. Additionally, the class Utils has all the functions needed for the program that are not related to the database.

Inside the model (orange), there are all the classes, which have an equivalent in the database (Fig. 3.2) as a table, meaning that the attributes in the tables and the ones inside the class are the same. Lastly, the view (green) contains all the classes with graphic

interfaces.

3.4 Implementation

3.4.1 Exercises and Evaluation

To manage the data of the application, we used sqlite3. We have chosen it, because in this program, we will not have a great quantity of data. Sqlite does not require installation or maintenance, it is just a single file that will be in a folder of the program and can be easily moved.
All recording and videos created and/or needed by the application are stored in a folder inside of the project, the structure of the folders is:
Resources->PatientName->SessioX->data.

To join the tables of the database with the classes in Java, we used Hibernate. To use it in our project, first we created a file called hibernate.cfg.xml. This is the configuration file. In this file, there needs to be the language of the database (sqlite in our case), the location of the database, the connection driver and all the mapping files of the project.

```
//Database language
 <property
     name="hibernate.dialect">org.hibernate.dialect.SQLiteDialect</property>
  //Connection driver
 <property name="hibernate.connection.driver_class">org.sqlite.JDBC</property>
 //Database location
 <property
     name="hibernate.connection.url">jdbc:sqlite:RememoryDatabase.db</property>

 //Mapping classes
 <mapping resource="model/Usuari.hbm.xml"/>
 ...
```

Mapping files, as the name suggests, are used to map the files and their attributes to the tables of the database. There needs to be one for each table of the database that we want as a class in our Java project.

Regarding the users, there are two types: administrator and evaluator. We decided not to create an inheritance as both have the same attributes and the same methods. So to distinguish them, we created a variable called rol; this variable will identify the user as an administrator if its value is one and as an evaluator if it is two. Regarding to the security of the application, the passwords are encrypted using SHA-256. Using this encryption there is no way to revert it, so to compare the passwords entered by the user, the program, compares the encryption of said password to the one in the database.

All the windows in the application are created using Java. In Java, each window is a JFrame, where we can put other components inside. To organize the different components

inside the frames, we used layouts, but different ones depending on the frame. The layouts used are:

- Grid Bag Layout: It is the most flexible of all the available layouts. It aligns the components by placing them in a grid of columns and rows. It allows the components to occupy more than one cell and to create margins on each component individually. Also, the height and width of each column and raw can be modified independently to the other cells.

- Border layout: Contrary to the anterior layout, border layout is one of the simplest, there are five areas to put the components (top, down, left, right and center). This is used for the general structure of the frame.

- Flow Layout: This layout just punts all components in a row, creating another if there is not enough space in the container. This layout is only used in the questionnaire (Fig. 3.7).

- Default layout: This layout is applied automatically when the Netbeans graphic interface editor is used. The advantage of this layout is that you can create an interface using drag and drop in the GUI editor making the creation of frames faster and more visually appealing. As a disadvantage, the code created from this layout can not be modified, and each component needs to be created manually and manually assigns their events. This is used for the simpler windows like figure 3.4.

To reproduce videos in a Java application, we used the library javaFX, that allows us to play videos inside a jFrame. The window of the visual test is done with a mix of Java swing and javaFX, the first one is used for creating the main frame and defines general structure of the window using a border layout. The second one allows to easily create a panel to show the video:

```
Media media = new Media(path);
 player = new MediaPlayer(media);
 MediaView view = new MediaView(player);
```

Once created the player, we just need to insert it into the frame, and javafx offers some useful components to help organize other components. These are VBox and HBox, which are respectively vertical and horizontal containers of other JavaFx components.
To create the recordings of this serious game, we used another library called javax. First, we need to establish the audioformat. We decided to set our audioformat with two channels, a sample of 8 bits and a sample rate of 6000 samples per second. This gave us enough quality to perfectly understand the recording without taking much space of the computer. After establishing the format, we need to create an audioline from where the voice will be recorded.

```
DataLine.Info info = new DataLine.Info(TargetDataLine.class, getAudioFormat());
TargetDataLine line = (TargetDataLine) AudioSystem.getLine(info);
```

And once created the line, we can start recording calling the start function from TargetDataLine. We also have to consider that the recording, when calling the start function, needs to be done from a new thread, as it is something that the computer needs to be constantly doing.
This thread will be closed upon stopping the recording or closing the current window. After recording it, the file is played in the window of the figure 3.8. To play it, we use JavaFX, using the same method mentioned early to create the video. In both cases to make the slider move on following the time of the player, we added an event to the slider that each time the current time of the player changes, the slider changed its value to the current time of the reproducer.

3.4.2 Memory enhancement

The algorithm of video description used for memory enhancement is an encoder-decoder for describing videos [1]. This algorithm uses:

- Convolutional Neural Networks (CNN): are designed to recognize visual patterns directly from pixel images. CNN consist of multiple layers of receptive fields. These fields are small neuron collections that process portions of the input image. The outputs of these collections are then tailed so that their input regions overlap to obtain better representation of the original image.

- Long Short-Term memory (LSTM): is a Recurrent neural network implemented in blocks. Each LSTM block contains three or four gates that control the flow of information into or out of their memory. These gates are implemented to compute a value between zero and one. Then, a multiplication with these values is applied to partially allow or deny information to flow in or out of the memory.

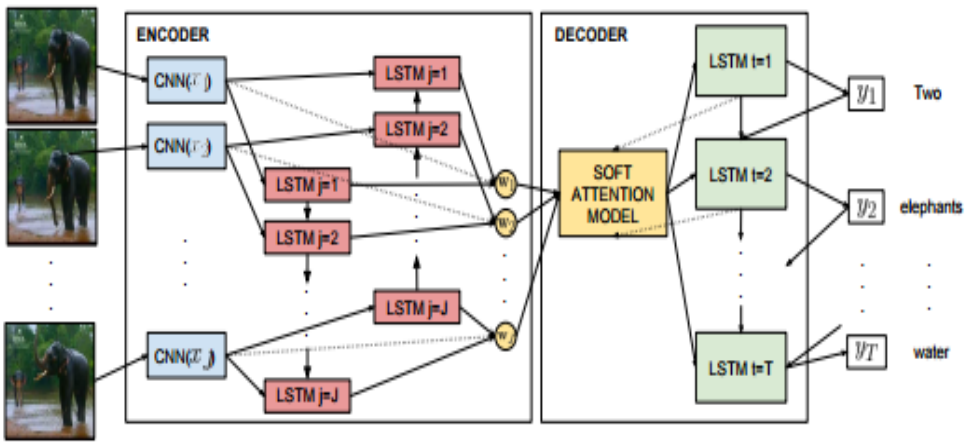

Figure 3.10: Encoder-Decoder: figure of the method presented in [1].

In figure 3.10, we can see the general structure of the algorithm. It consists of four stages using CNNs and LSTMs for describing images and modeling their temporal relationship, respectively. This model receives as input images or frames of a video. First of all, CNN models (blue boxes) are applied to the images to extract the feature vector of each image that gives us information about the objects appearing and their surroundings. This does not give us information about actions and relations along time, that is what LSTM (Long Short-Term Memory, red boxes) do. Applying forward and backward LSTM models with different weight matrices, estimated from the training set, we can solve this problem. Each feature vector is computed by concatenating the outputs of forward and backward LSTMs. Next, already in the decoder part, a soft attention model (yellow) receives the concatenation of the CNN and LSTM as input. This part decides which parts of the input images should focus for emitting the next word. Lastly, an LSTM (green in the image) generates the description from the data obtained from the other stages of the software. The length is obtained by using a softmax function on top of this LSTM.

To create a video from egocentric images generated by the wearable camera, we used two different algorithms one for segmentation [12] and one for summarization [13].

The segmentation algorithm [12] addresses the problem of organizing egocentric photo streams acquired by a wearable camera into semantically meaningful segments. Using CNNs, contextual and semantic information are extracted for each image. Afterwards, a vocabulary of concepts is defined in a semantic space. Finally, images with similar contextual and semantic attributes are grouped together.

The second and last algorithm to create the video is summaritazation[13], that automatically summarizes the egocentric photo streams. With a new CNN-based filter, all non-informative images are removed, then images are ranked by relevance to ensure semantic diversity and later re-ranked again by a novelty criterion to reduce redundancy.

Both of these algorithms were applied to the original egocentric images and with their output and adding music to it, we created a video with format .mp4. The obtained result is shown to the patients during their treatment in the 'Video' section of our treatment platform.

Chapter 4

Results and evaluation

4.1 Exercises

During the development of the EgoMemory treatment platform, we received the advice and opinions of an expert in usability. Following the tips given, we tried to separate as much as possible the parts where the doctor is alone and the parts where he or she is with the patient, while also following the design wanted for the doctors. The videos also needed to be bigger in relation to all other components of the interface, so the patient does not get distracted (a common situation with patients suffering from MCI) easily. All recordings created from the application are saved following a naming pattern: data_hour.wav and inside the folder of the patient, to try to keep an easy to follow organization.

4.2 Evaluation

The project is scheduled for real clinic tests on patients in April, 2017. So we do not have real results of the evaluation done by the doctor yet; although we tried to follow as much as we could the design of the doctors while implementing all functions needed.

4.3 Memory enhancement

When generating our model for video description, in order to get a quicker convergence of the CNN model weights, we applied a pre-training of the model on the Microsoft Research Video Description Corpus (MSVD) [3], which consists of 1970 open domain clips collected from YouTube. Each video has a variable number of descriptions written by different people, reaching more than 80,000 training samples.

After that, we applied a fine-tuning on the EDUB-SegDesc dataset, which is composed of egocentric sequences (see section 4.3.1). Although the problem tackled by both datasets is similar, we must consider the following differences:

1. The tense: the descriptions in our dataset need to be in past tense: most of the sentences of the ground truth from the original dataset are in present, while in our project they need to be in past, as egocentric images are caption from past experiences of the patient.

2. The point of view: the MSVD dataset uses YouTube videos as a sample for the algorithm descriptions; most of the videos are made in third person, so you can see the person filming, while our project needs to work with egocentric images that are taken from the point of view of the person wearing the camera. So the captions generated should be in first person.

The MSVD dataset, on all sets, was trained using a batch size of 64, the learning rate was automatically set by the Adadelta [15] method, and to reduce the computational load, only one image of every 26 frames was picked. The hyperparameters were randomly set, such hyperparameters and their ranges are m -> [300, 700], |ht| -> [1000, 3000], being m the size of the word embedding and |ht| the hidden state. When using the BLSTM encoder, we performed an additional selection on |vj | -> [100, 2100], being |vj| each feature vector.

4.3.1 Dataset

We created a new dataset, that we will call Egocentric Dataset of the University of Barcelona-Segments Description (EDUB-SegDesc, for short). To train this dataset, so that it can get acceptable results with the egocentric images, we had to create a sample of egocentric images, and train this new sample together with the best model of the ABiViRNet dataset.
As we applied fine-tuning of the previous model, the hyperparameters were maintained at their previous values when we retrained ABiViRNet with the dataset EDUB-SegDesc.
First of all, to train the dataset we needed enough data, so from a sample of 37430 images of 45 days from 8 people, we divided it in 1140 segments and generated 3261 descriptions, 3 descriptions per segment.

```
//Examples of the descriptions of two video segments
-First segment
I worked on my office
I worked with my laptop
I was in my office working
-Second segment
I saw two men in a room
I used my phone while having lunch
I ate in a white room
```

Example 1

Figure 4.1: Different images from different segments

In figure 4.1, we can see the complexity of these egocentric images, as they are all really different, some are darker, some are clearer, and in a large variety of situations.

After gathering and annotating the dataset, we separated it in three parts:

1. 80% used for training, used by the algorithm to learn and create the matrices of the encoder. This part consists in 871 segments, each with a variable number of images.
2. 15% used as test, the software uses this to get the final result and compare all different models. In this case, we got 187 segments.
3. 5% used as validation; this part is used to search for the best model of the generated ones through the algorithm. Used as values there were 82 segments.

This separation was made by randomly choosing days of the initial sample and putting them in random parts until the percentage was fulfilled.

Once the initial data was created and distributed in the right folders, we had to modify the configuration file of the project to accept new inputs and to train on top of the already trained project. In addition to modifying the config file, we also needed to add the trained vocabulary of the MSVD dataset in the project.

```
// Adding vocabulary
/*load the pretrained model*/
```

```
dataset_pretrained =
    kw.loadDataset('trained_models/Bes_Model/Dataset_MSVD_features.pkl')

/*substitutes old vocabulary for the one of the pretrained model */
dataset.vocabulary_len = dataset_pretrained.vocabulary_len
dataset.n_classes_text = dataset_pretrained.n_classes_text
```

Furthermore, we also trained our model without applying a pre-training on the MSVD dataset in order to compare how it effects its performance.

4.3.2 Metrics

In order to evaluate and compare the results of the different models, the standardized COCO-Caption evaluation package[4] was used, which provides metrics for text description comparison. The main metrics used was:

BLEU[5]: It is a metric that compares the ratio of n-gram structures that are shared between the ground truth and the hypotheses created by the system.

More specifically, in our system we compared the results of Bleu-4. This result will be a number between 0 and 1, and indicates how similar the candidate and reference texts are, with values closer to 1 representing more similar texts.

4.3.3 Experimental results

After loading the corresponding vocabulary, we started training. The training was made with a batch size of 64 and an epoch maximum of 50. We also set the configuration to save the models of all epochs to not miss any information. Additionally, we declared patience as ten. This parameter stops the execution if the metric Bleu-4 does not improve in this number of evaluations. In our training, the evaluations stopped at epoch 41, as the metric Blue-4 did not improve since epoch 30.

```
//Execution line from the console
Epoch 41: early stopping. Best Bleu-4 value found at epoch 30: 0.3493
```

The results of the evaluation were:

The results in table 4.1, are the best calculated by the algorithm with our dataset, corresponding at epoch 30.
As we can see in the table, our results are worse than the original ones (see table above) on all metrics. The division of the table separates the results of the MSVD (first row) from the ones of the EDUB-SegDesc (last two). As we can see in the table, the metric BLEU-4_gotten in [1] is far above the results in our execution, that is because the descriptions created for the egocentric images are different from the ones of the original dataset (past tense and person). This also means that the impact of training on top of the model MSVD is not that big; the metric just increased in 2.2 points.

Note that our metric measures the similarity of a generated sentence against a set of ground truth sentences written by humans. Even tough the difference is quite significant,

Model	BLEU4[%]
ABiViRNet	53.6
EDUB-SegDes + MSVD	34.9
EDUB-SegDes	32.2

Table 4.1: Results of the different models

our results were even better then expected, as our sample of 3261 descriptions is really small compared to the original one (80000).

4.3.4 Prediction results

In this section, we show and discuss some result examples of the algorithm. The data are three random images taken from one segment. Below each photo, we can see the ground truth sentences, created manually, followed by Egocentric + MSVD. This caption is the result obtained for the segment through the [1] algorithm pre-trained on ABiViRNet and fine-tuned on EDUB-SegDesc. And the last line shows the prediction obtained by the same model trained from scratch on the EDUB-SegDesc dataset. More examples are shown in the Appendix.

Example 1:

Figure 4.2: Example of egocentric images from the same segment

- Ground truth: I drove my car / I used my car / I travelled by car
- EDUB-SegDesc + MSVD: I went to a car
- EDUB-SegDesc: I was at the street

In this example, we can see that the result from Egocentric + MSVD is far superior than the egocentric one, it can detect the movement and that the person wearing the camera is in or near a car, while the egocentric one can not identify the car and disregards the

parts where it is in an interior. This can be explained by the lack of more samples in the egocentric dataset.

Example 2:

Figure 4.3: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I walked on the street / I walked in a parking / I walked in a city
- EDUB-SegDesc + MSVD: i walked on the street
- EDUB-SegDesc: i walked on the street

In this example, both algorithms obtain the same description that is also the same as one of the ground truth. Both are correct; this is an easy segment to detect, because all images are clean and many of the samples created in the dataset are of people walking on the street.

Example 3:

Figure 4.4: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I talked to a woman/ I bought something from a woman/I woman sold me something to eat
- EDUB-SegDesc + MSVD: I was in a bar
- EDUB-SegDesc: I was in a bar with other people

This is case where the result of both models fail. It is an uncommon situation (in a market buying something). So in the dataset, there were no many images similar to the ones in this segment. But if we take a closer look at both descriptions and at the segments images, we can see that they are not that far off. They detect a person behind a bar with food and in a crowded place, which are similar characteristics of a bar.

Chapter 5

Conclusions

In this project, we have created a new image-based tool to ease the work of the medical staff on their job of diagnosing and treating people suffering from MCI, EgoMemory. EgoMemory consists on a java application that implements a serious game, it has a logging system with different type of users users and also has different patients, each with its own data gotten from the different tests and from the serious game. This serious game is created using video description algorithms and applying state of the art technologies like Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BLSTM). Using the egocentric images and the dataset ABiViRNet[1] we created a new model for generating captions from these images and we also created videos from the egocentric images using segmentation[12] and summaritzation[13]. We also validated that the results of Video Description are quite similar to the ground truth, failing on some cases due to the small sample of the dataset EDUB-SegDesc in comparasion to the ABiViRNet[1].

One important part of EgoMemory is the part of creating a video description from egocentric images. Generating video descriptions from (conventional) images is a very recent trend in Computer Vision, that represents an interesting, but hard Computer Vision and Machine Learing task. In this project, we worked on a really complex and novel problem, using video description but on egocentric images. Its difficulty comes from the small information of the images (egocentric images have much narrower field of view) and the timespan between them (there is almost no temporal coherence between consecutive egocentric images); this is what makes it hard for the algorithm to generate the correct captions for a segment of images. Even though facing these difficulties and the relatively small sample available (in comparison to the original MSVD), in most cases the caption generated was the same or nearly the same as the ground truth. To work with the video description algorithm, we used CNNs for object recognition, and also a LSTM that acts as a language model to generate the sentence describing the images.

The program in conjunction with [2] will provide the medical community a novel image-based tool that will allow for a faster and better treatment and diagnose of neurological illnesses, proportioning a serious game that will allow to exercise the memory of the patients and test to improve the follow-up of neurological illnesses.

Future Work

This project does not end here; there is still improvements to be made and more functions to add. One part of the project is to add visual games and an emotion test to each episode in the visual tests. Those tests will improve the capacity of the doctor to detect and track the MCI progress on the patient. Another task of the project is to add the patients textual tests answers to the database. It is a hard task and it needs time to be done.

In addition, we plan to work on developing a metric for offering an objective comparison of the descriptions provided by the patients and the descriptions provided by our algorithm, Which will allow a further automation of the treatment and the evaluation procedure.

Bibliography

[1] Peris Á, Bolaños M, Radeva P, Casacuberta F. Video description using bidirectional recurrent neural networks. In International Conference on Artificial Neural Networks 2016 Sep 6 (pp. 3-11). Springer International Publishing.

[2] J. Sánchez (2016). ReMemory: Sistema d'Avaluació per al Tractament de Persones amb Deteriorament Cognitiu Lleu. Memory of Final Grade Degree, Departament de Matemática Aplicada i Anàlisi, UB .

[3] Chen, D. L., and Dolan, W. B. (2011, June). Collecting highly parallel data for paraphrase evaluation. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 190-200). Association for Computational Linguistics.

[4] Chen, X., Fang, H., Lin, T. Y., Vedantam, R., Gupta, S., DollÃ¡r, P., and Zitnick, C. L. (2015). Microsoft COCO captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325.

[5] Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. (2002, July). BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting on association for computational linguistics (pp. 311-318). Association for Computational Linguistics.

[6] Gauthier, S., Reisberg, B., Zaudig, M., Petersen, R. C., Ritchie, K., Broich, K., and Cummings, J. L. (2006). Mild cognitive impairment. The Lancet, 367(9518), 1262-1270.

[7] Byun, S., and Park, C. (2011, July). Serious game for cognitive testing of elderly. In International Conference on Human-Computer Interaction (pp. 354-357). Springer Berlin Heidelberg.

[8] Manera, V., Petit, P. D., Derreumaux, A., Orvieto, I., Romagnoli, M., Lyttle, G., ... and Robert, P. H. (2015). 'Kitchen and cooking,' a serious game for mild cognitive impairment and Alzheimer's disease: a pilot study.

[9] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell: A neural image caption generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3156-3164).

[10] Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., and Courville, A. (2015). Describing videos by exploiting temporal structure. In Proceedings of the IEEE international conference on computer vision (pp. 4507-4515).

[11] Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. (2015). Sequence to sequence-video to text. In Proceedings of the IEEE International Conference on Computer Vision (pp. 4534-4542).

[12] Dimiccoli, M., Bolaños, M., Talavera, E., Aghaei, M., Nikolov, S. G., and Radeva, P. (2016). SR-clustering: Semantic regularized clustering for egocentric photo streams segmentation. Computer Vision and Image Understanding, Volume 155, February 2017, Pages 55–69.

[13] Lidon, A., Bolaños, M., Dimiccoli, M., Radeva, P., Garolera, M., and Giró-i-Nieto, X. (2015). Semantic Summarization of Egocentric Photo Stream Events. arXiv preprint arXiv:1511.00438.

[14] Prince, M., Guerchet, M., and Prina, M. (2013). The global impact of dementia 2013-2050. Alzheimer's Disease International, ALZHEIMER'S DISEASE INTERNATIONAL, THE GLOBAL VOICE ON DEMENTIA.

[15] Zeiler, M. D. (2012). ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.

[16] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. (2014). Sequence to sequence learningwith neural networks. In Advances in Neural Information Processing Systems, volume 27 (pp. 3014-3112).

[17] Petersen, R. C. (2016). Mild cognitive impairment. CONTINUUM: Lifelong Learning in Neurology, 22(2, Dementia), 404-418.

[18] Bolaños M, Dimiccoli M, Radeva P. Toward Storytelling From Visual Lifelogging: An Overview. IEEE Transactions on Human-Machine Systems. 2016 Oct 27.

Appendix

Prediction Results

good examples

Figure 5.1: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I walked on the street and entered in a building/ I walked on the street/ I went to a university
- EDUB-SegDesc + MSVD: I walked on the street
- EDUB-SegDesc: I walked on the street

In this case, both models, correctly guess the sentence, as in the sample there are many segments of people just walking on the street. 'University' or 'entered in a building' are not that common.

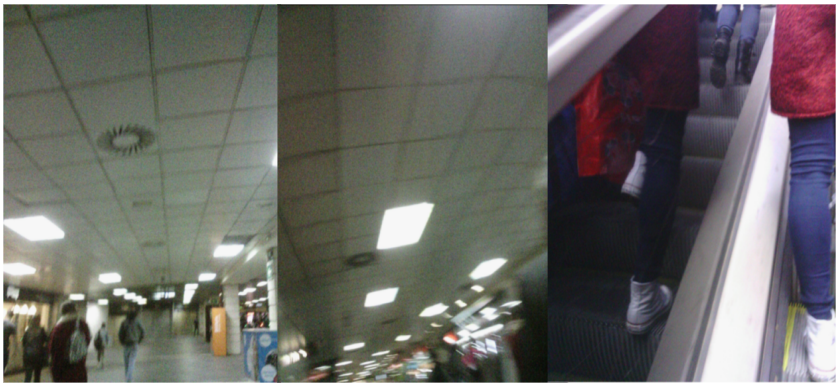

Figure 5.2: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I left the station/ I walked in the station/ I walked and used automatic stairs
- EDUB-SegDesc + MSVD: i was on the train
- EDUB-SegDesc: i went to the bathroom

This example is an other that the results are nos that good, that is because of the little information that this segments gives, as this segment only has three egocentric images. Even tough, the EDUB-SegDesc + MSVD model, cant detect that the action is happening in a public place.

Figure 5.3: Example of egocentric images from the same segment

Ground truth:

- Ground truth:I drove a car/ I travelled by car/ I stepped in the car and drove it
- EDUB-SegDesc + MSVD: i was in a car

- EDUB-SegDesc: i was in a car

In this example, both models get the same description, and even tough it is not the same as any of the ground truths, is quite similar to them, as both of them detect the car. The descriptions are true but lack information, that can be due to the small number of images in the dataset EDUB-SegDesc.

Figure 5.4: Example of egocentric images from the same segment

Ground truth:

- Ground truth:I went to the bathroom /I walked to the bathroom
- EDUB-SegDesc + MSVD: i went to the bathroom
- EDUB-SegDesc: i went to the bathroom

bad examples

Figure 5.5: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I saw a boxing match/ I watch a boxing match while sitting on a chair/I looked at a boxing match

- EDUB-SegDesc + MSVD: I was in a bar

- EDUB-SegDesc: I was in a conference

This segment is a really difficult one, as it is the only one with a boxing match, and it is not a common scenario. Although in both cases it seems to get that they are in a place with people, and in this case the EDUB-SegDesc can even detect that the people are watching something or someone.

Figure 5.6: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I was inside a train/ I travelled by train / I was in a train

- EDUB-SegDesc + MSVD: I was in a house

- EDUB-SegDesc: I was in a restaurant

The images in this fragment are not very clear for the models, they identify the color white from the background and the people, but are quite far away from the original descriptions.

Figure 5.7: Example of egocentric images from the same segment

Ground truth:

- Ground truth: I walked on the street and entered in a building /I Walked on the street and went inside a boxing building/ I was on the street and went inside a boxing match
- EDUB-SegDesc + MSVD: I was in a bar
- EDUB-SegDesc: I was in a bar with other people

This example is similar to the first one, the situation is not really common, as the photos are made at night time with light coming from the shops. In this case, we can see that the models can detect that the photos are taken in a place with people and food or drinks, but again 'boxing match' is not a common word in the descriptions of the samples.